LPC1700 系列微控制器

第 13 章 USB OTG 控制器

用户手册 Rev00.04

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4

网址：http://www.zlgmcu.com

销售与服务网络

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4 邮编：510630

电话：(020)38730972 38730976 38730916 38730917 38730977

传真：(020)38730925

网址：http://www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室

电话：(020)87578634 87569917

传真：(020)87578842

南京周立功

地址：南京市珠江路 280 号珠江大厦 2006 室

电话：(025)83613221 83613271 83603500

传真：(025)83613271

北京周立功

地址：北京市海淀区知春路 113 号银网中心 A 座 1207-1208 室（中发电子市场斜对面）

电话：(010)62536178 62536179 82628073

传真：(010)82614433

重庆周立功

地址：重庆市石桥铺科园一路二号大西洋国际大厦（赛格电子市场）1611 室

电话：(023)68796438 68796439

传真：(023)68796439

杭州周立功

地址：杭州市天目山路 217 号江南电子大厦 502 室

电话：(0571)89719480 89719481 89719482 89719483 89719448 89719485

传真：(0571) 89719494

成都周立功

地址:成都市一环路南二段 1 号数码同人港 401 室(磨子桥立交西北角)

电话：(028) 85439836 85437446

传真：(028) 85437896

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 C 座 4 楼 D 室

电话：(0755)83781788（5 线）

传真：(0755)83793285

武汉周立功

地址:武汉市洪山区广埠屯珞瑜路 158 号 12128 室(华中电脑数码市场)

电话：(027)87168497 87168297 87168397

传真：(027)87163755

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室

电话：(021)53083452 53083453 53083496

传真：(021)53083491

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室

电话：(029)87881296 83063000 87881295

传真：(029)87880865

目录

第13章 USB OTG控制器

13.1 如何阅读本章

本章描述了 LPC1768、LPC1766、LPC1765、LPC1758、LPC1756 和 LPC1754 可以使用的 USB OTG 控制器。

13.2 基本配置

利用下列寄存器来配置 USB 控制器：

a）功率：在寄存器 PCONP 中置位 PCUSB。

注：复位后，USB 模块被禁止（PCUSB=0）。

b）时钟：USB 模块可与专门的 PLL 或主 PLL 一起使用来获得 USB 时钟（请参见“PLL1 寄存器描述”小节）。

c）引脚：分别在寄存器 PINSEL0-5 和寄存器 PINMODE0-5 中选择 USB 引脚和引脚的模式（请参见“引脚连接模块”章节的“寄存器描述”小节）。

d）唤醒：USB 总线端口上的活动可将微控制器从掉电模式中唤醒（请参考“掉电模式支持”小节和“从低功耗模式中唤醒”小节）。

e）中断：利用相应的中断置位使能寄存器使能 NVIC 中的中断。

f）初始化：请参见“USB 控制器的初始化”小节。

13.3 简介

本章描述了 OTG 和 USB 2.0 OTG 双角色设备控制器的 I^2C 部分，该 OTG 控制器集成了（OHCI）主机控制器、设备控制器和 I^2C。I^2C 接口（仅为主机）控制外部 OTG 收发器。

USB OTG 是 USB2.0 规范的补充，它通过增加连接到 USB 外设的主机功能来扩充现有移动设备和 USB 外设的能力。USB2.0 规范和 USB OTG 的信息可在 USB 执行论坛上找到。

13.4 特性

- 完全遵循 USB2.0 的补充规范：USB OTG V1.0a；
- 硬件支持主机交换协议（HNP）；
- 包括 HNP 和 SRP（对话请求协议）所需的可编程的定时器；
- 支持任何遵循 OTG 收发器规范（CEA-2011）V1.0 的 OTG 收发器。

13.5 结构

USB OTG 控制器的方框图如下图所示。

主机、设备、OTG 和 I^2C 控制器可通过寄存器接口进行编程。OTG 控制器可通过 HNP 协议使能主机和设备之间的动态切换。可将一个端口连接到外部的 OTG 收发器来支持 OTG 连接。寄存器接口和外部 OTG 收发器之间的通信可通过 I^2C 接口和外部 OTG 收发器来处理。

对于仅使用设备或主机控制器（不是 OTG）的 USB 连接，端口使用一个内置的 USB 模拟收发器（ATX）。

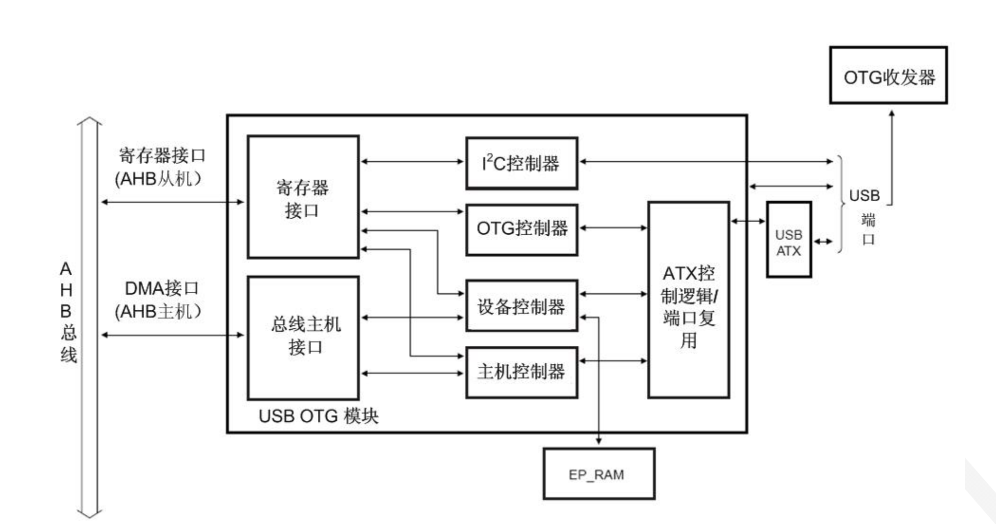


图 13.1　USB OTG 控制器方框图

13.6 操作模式

OTG 控制器可以在下面的模式中操作：

- 主机模式（见图 13.2）；
- 设备模式（见图 13.3）；
- OTG模式（见图 13.4）。

13.7 引脚配置

OTG 控制器只有 1 个 USB 端口。

表 13.1　USB OTG 端口的引脚

引脚名称	方向	描述	引脚类别
USB_D+	I/O	正向差分数据	USB 连接器
USB_D-	I/O	负向差分数据	USB 连接器
USB_UP_LED	O	Good Link LED 控制信号	控制
USB_SCL	I/O	I^2C 串行时钟	外部 OTG 收发器
USB_SDA	I/O	I^2C 串行数据	外部 OTG 收发器

以下各图给出了实现连接到 USB 设备的不同方法。这里描述的例子使用 ISP1301（NXP）用于外部 OTG 收发器和 USB 主机电源开关 LM3526-L（国家半导体）。

13.7.1 连接USB端口到外部OTG收发器

对于OTG功能，必须将OTG收发器连接到LPC1700 系列Cortex-M3 微控制器设备：使用USB信号的内部USB收发器，并仅使用OTG功能的外部OTG收发器（见图 13.2）。该选项在VP/VM模式下使用内部收发器。

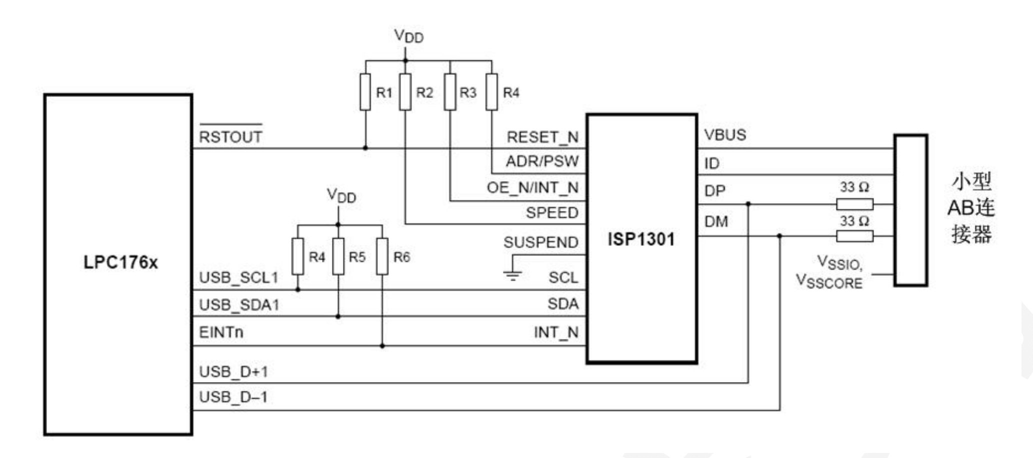


图 13.2 USB OTG 的端口配置

13.7.2 将USB作为主机的连接

利用一个嵌入的 USB 收发器可将 USB 端口作为主机进行连接。该端口不具有 OTG 功能。

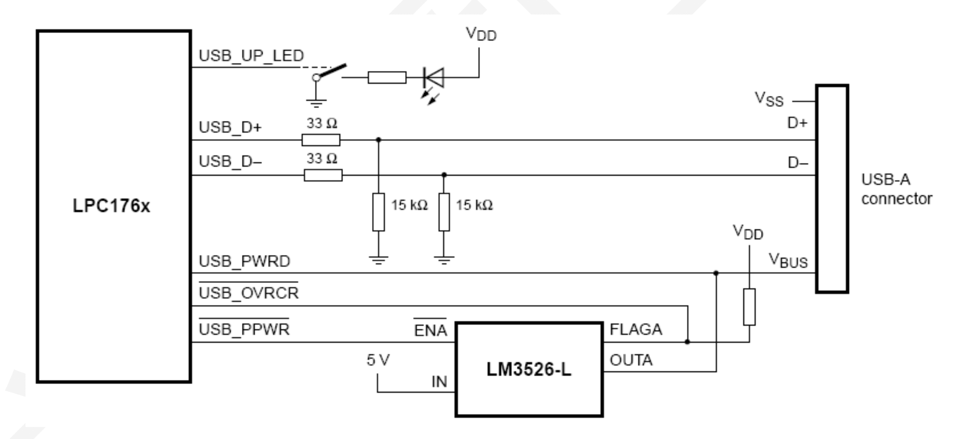


图 13.3 USB 主机端口的配置

13.7.3 将USB作为设备的连接

USB 端口可作为设备进行连接。该端口不具有 OTG 功能。

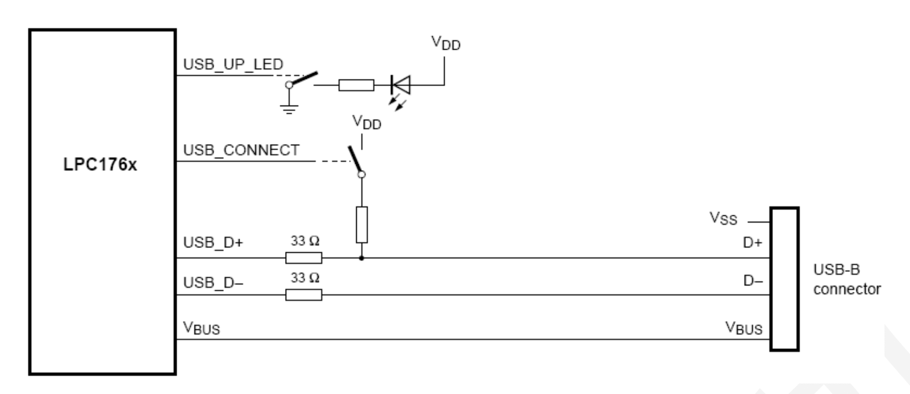


图 13.4 USB 设备端口的配置

13.8 寄存器描述

下表对 OTG 和 I^2C 寄存器进行了总结。

设备和主机寄存器分别在“USB 设备控制器”章节的和“USB 主机控制器”章节中描述。所有的寄存器为 32 位宽并以字地址边界排列。

表 13.2 USB OTG 和 I^2C 寄存器地址的定义

名称	地址	访问	功能
中断寄存器			
USBIntSt	0x400F C1C0	R/W	USB 中断状态
OTG 寄存器			
OTGIntSt	0x5000 C100	RO	OTG 中断状态
OTGIntEn	0x5000 C104	R/W	OTG 中断使能
OTGIntSet	0x5000 C108	WO	OTG 中断设置
OTGIntClr	0x5000 C10C	WO	OTG 中断清除
OTGIntCtrl	0x5000 C1110	R/W	OTG 状态和控制
OTGTmr	0x5000 C1114	R/W	OTG 定时器
I^2C 寄存器			
I2C_RX	0x5000 C300	RO	I^2C 接收
I2C_TX	0x5000 C300	WO	I^2C 发送
I2C_STS	0x5000 C304	WO	I^2C 状态
I2C_CTL	0x5000 C308	R/W	I^2C 控制
I2C_CLKHI	0x5000 C30C	R/W	I^2C 时钟高电平
I2C_CLKLO	0x5000 C310	WO	I^2C 时钟低电平
时钟控制寄存器			
OTGClkCtrl	0x5000 CFF4	R/W	OTG 时钟控制器
OTGClkSt	0x5000 CFF8	RO	OTG 时钟状态

13.8.1 USB中断状态寄存器（USBIntSt-0xE01F C1C0）

USB OTG 控制器有 7 条中断线。该寄存器允许软件对其执行一次读操作来确定这 7 条中断线的状态。

将 7 条中断线相“或”，连接到向量中断控制器的一个通道中。

表 13.3 USB 中断状态寄存器位描述

位	符号	描述	复位值
0	USB_INT_REQ_LP	低优先级中断线的状态。该位只读	0
1	USB_INT_REQ_HP	高优先级中断线的状态。该位只读	0
2	USB_HOST_REQ_DMA	DMA 中断线的状态。该位只读	0
3	USB_HOST_INT	USB 主机中断线的状态。该位只读	0
4	USB_ATX_INT	外部 ATX 中断线的状态。该位只读	0
5	USB_OTG_INT	OTG 中断线的状态。该位只读	0
6	USB_I2C_INT	I^2C 模块中断线的状态。该位只读	0
7	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA
8	USB_NEED_CLK	USB 所需时钟指示器。该位只读	1
30：9	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA
31	EN_USB_INTS	使能所有 USB 中断。当该位清零时，向量中断控制器看不到 USB 中断线的“或”输出	1

13.8.2 OTG中断状态寄存器（OTGIntSt-0xE01F C100）

当中断事件在 HNP 主机交换序列过程中出现时，该寄存器中的位由硬件置位。有关这些位何时置位的更多信息请参考“支持 HNP”小节。

表 13.4 OTG 中断状态寄存器位描述

位	符号	描述	复位值
0	TMR	定时器超时	0
1	REMOVE_PU	移除上拉电阻	0
2	HNP_FAILURE	HNP 失败 该位由硬件置位来表示 HNP 转换失败	0
3	HNP_SUCCESS	HNP 成功 该位由硬件置位来表示 HNP 转换成功	0
31：4	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA

13.8.3 OTG中断使能寄存器（OTGIntEn-0x5000 C104）

向该寄存器中的位写入 1 使 OTGIntSt 中的相应位在其中一条中断线上产生中断。该中断与 USBIntSt 寄存器中的 USB_OTG_INT 中断线相连。OTGIntSt 寄存器的位分配以及复位值与 OTGIntSt 寄存器相同。

13.8.4 OTG中断置位寄存器（OTGIntSet-0x5000 C20C）

向该寄存器的位写入 1 将置位 OTGIntSt 寄存器中的相应位。写入 0 无效。OTGIntSet 寄存器的位分配与 OTGIntSt 寄存器相同。

13.8.5 OTG中断清除寄存器（OTGIntClr-0x5000 C10C）

向该寄存器的位写入 1 将清零 OTGIntSt 寄存器中的相应位。写入 0 无效。OTGIntClr 寄存器的位分配与 OTGIntSt 寄存器相同。

13.8.6 OTG状态和控制寄存器（OTGStCtrl-0x5000 C110）

OTG 状态和控制寄存器允许在 HNP 移交序列过程中使能硬件跟踪、控制 OTG 定时器、监控定时器计数并控制映射到端口 U1 到 U2 的函数。

对时间要求严格的事件在切换序列中通过 OTG 定时器来控制。定时器可在下面两种模式下操作 ：

- 单次触发模式：中断在 TIMEOUT_CNT 的结束产生（见“OTG 定时器寄存器”小节），OTGIntSt 中的 TMR 置位，并且定时器将被禁能；
- 自由运行模式：中断在 TIMEOUT_CNT 的结束产生（见“OTG 定时器寄存器”小节），TMR 置位，并且定时器值被重新载入计数器中。定时器在该模式中不被禁能。

表 13.5 OTG 状态控制寄存器位描述

位	符号	描述	复位值
1:0	PORT_FUNC	控制端口的函数。当 B_HNP_TRACK 或 A_HNP_TRACK 置位且 HNP 成功移交时，Bit0 通过硬件置位或清零。请见“支持 HNP”小节。Bit1 保留	-
3:2	TMR_SCALE	定时器调节选择。该字段确定了每个定时器的持续时间 00：10μs（100KHz） 01：100μs（10KHz） 10：1000μs（1KHz） 11：保留	0x0
4	TMR_MODE	定时器模式选择 0：单次触发模式 1：自由运行模式	0
5	TMR_EN	定时器使能。该位置位时，TMR_CNT 加 1。该位清零时，TMR_CNT 复位为 0	0
6	TMR_RST	定时器复位。向该位写入 1 会将 TMR_CNT 复位为 0。这可以使软件对某个位进行单独控制以在定时器使能时重启定时器	0
7	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA
8	B_HNP_TRACK	使能 B-设备（外设）的 HNP 跟踪，请参见“支持 HNP”小节 当 HNP_SUCCESS 或 HNP_FAILYRE 置位时，硬件会清零该位	0
9	A_HNP_TRACK	使能 A-设备（主机）的 HNP 跟踪，请参见“支持 HNP”小节 当 HNP_SUCCESS 或 HNP_FAILYRE 置位时，硬件会清零该位	0

续上表

位	符号	描述	复位值
10	PU_REMOVED	当 B-器件从外设变为主机时，软件在其移除 D+上拉时将该位置位，请参见“支持 HNP”小节。当 HNP_SUCCESS 或 HNP_FAILYRE 置位时，硬件会清零该位	0
15:11	-	保留，用户软件不向保留位写入 1，从保留位读取的值未被定义	NA
31:16	TMR_CNT	当前定时器的计数值	0x0

13.8.7 OTG定时器寄存器（OTGTmr-0x5000 C114）

表 13.6 OTG 定时器寄存器位描述

位	符号	描述	复位值
15:0	TIMEOUT_CNT	TMR 中断在计数值达到该值时置位	0xFFFF
31:16	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA

13.8.8 OTG时钟控制寄存器（OTGClkCtrl-0x5000 CFF4）

该寄存器控制 OTG 控制器的时钟。只要软件想访问寄存器，就必须置位相应的时钟控制位。如果相对应的 OTGClkCtrl 位已经置位，则软件无需在每次访问寄存器时都重复该操作。

表 13.7 OTG 时钟控制寄存器位描述

位	符号	值	描述	复位值
0	HOST_CLK_EN		主机时钟使能	0
		0	禁能主机时钟	
		1	激活主机时钟	
1	DEV_CLK_EN		设备时钟使能	0
		0	禁能设备时钟	
		1	激活设备时钟	
2	I2C_CLK_EN		I^2C 时钟使能	0
		0	禁能 I^2C 时钟	
		1	激活 I^2C 时钟	
3	OTG_CLK_EN		OTG 时钟使能	0
		0	禁能 OTG 时钟	
		1	激活 OTG 时钟	
4	AHB_CLK_EN		AHB 主机时钟使能	0
		0	禁能 AHB 主机时钟	
		1	激活 AHB 主机时钟	
31:5	-	NA	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA

13.8.9 OTG时钟状态寄存器（OTGClkSt-0x5000 CFF8）

该寄存器保存时钟的有效状态。当通过 OTG 时钟控制寄使能时钟时，软件应该查询该寄存器中相应的位。如果该位置位，则软件可通过寄存器访问继续操作。如果 OTGClkCtrl 位没有改变，则软件无需在每次访问寄存器时都重复该操作。

表 13.8 OTG 时钟状态寄存器位描述

位	符号	值	描述	复位值
0	HOST_CLK_ON		主机时钟的状态	0
		0	主机时钟不可用	
		1	主机时钟可用	
1	DEV_CLK_ON		设备时钟的状态	0
		0	设备时钟不可用	
		1	设备时钟可用	
2	I2C_CLK_ON		I^2C 时钟的状态	0
		0	I^2C 时钟不可用	
		1	I^2C 时钟可用	
3	OTG_CLK_ON		OTG 时钟的状态	0
		0	OTG 时钟不可用	
		1	OTG 时钟可用	
4	AHB_CLK_ON		AHB 主机时钟的状态	0
		0	AHB 时钟不可用	
		1	AHB 时钟可用	
31:5	-		保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	NA

13.8.10 I^2C接收寄存器（I2C_RX-0x5000 C300）

该寄存器是接收 FIFO 的顶部字节。接收 FIFO 的深度为 4 字节。RxFIFO 通过硬复位或软复位（I2C_CTLbit7）来刷新。读取空 FIFO 会得到不可预知的数据结果。

表 13.9 I^2C 接收寄存器位描述

位	符号	描述	复位值
7:0	RX_DATA	接收数据	-

13.8.11 I^2C发送寄存器（I2C_TX-0x5000 C300）

该寄存器是发送 FIFO 的顶部字节。发送 FIFO 的深度为 4 字节。

Tx FIFO 通过硬复位或软复位（I2C_CTLbit7）或在仲裁失败出现（I2C_STS bit3）时被刷新。写入满 FIFO 的数据被忽略。

必须写 I2C_TX 进行读写操作来传输每个字节。主机接收操作时 Bit7[7:0]被忽略。主机接收器必须为它期望在 RX FIFO 中接收到的每个字节写一个伪字节到 Tx FIFO。当停止位置位或起始位置位以使在写入 TX FIFO 的字节上产生重启条件时，从从机读取的字节不会被应答。也就是主机接到的最后一个字节不会被应答。

表 13.10 I^2C 发送寄存器位描述

位	符号	描述	复位值
7:0	TX DATA	发送数据	-
8	STRAT	为 1 时，在发送该字节前发送一个起始条件	-
9	STOP	为 1 时，在发送完该字节后发送一个停止条件	-
31:10	-	保留，用户软件不应向保留位写入 1，从保留位读取的值未定义	-

13.8.12 I²C状态寄存器（I2C_STS-0x5000 C304）

I2C_STS 寄存器提供了 TX 和 RX 模块的状态信息以及外部总线的当前状态。各个位通过 I2C_CTL 寄存器使能为中断并连接到 USBIntSt 的位 I2C_USB_Int。

表 13.11 I²C 状态寄存器位描述

位	符号	值	描述	复位值
0	TD		处理结束中断。处理成功完成时该位置位。向状态寄存器的位 0 写入 1 可清除中断。从机操作对其没有影响	0
		0	处理还没有完成	
		1	处理已完成	
1	AFI		仲裁失败中断。发送时，若 SDAOUT 为高时 SDA 为低，I²C 会失去对另一个总线设备的仲裁，这时仲裁失败位置位。向状态寄存器的位 1 写入 1 可清除中断	0
		0	在最后一次发送时没有仲裁失败	
		1	在最后一次发送时出现仲裁失败	
2	NAI		无应答中断。每发送完一个字节后，发送器都希望从接收器那接收一个应答。若没有收到应答，该位置位。当主机 TX FIFO 有字节写入时中断被清除	0
		0	发送完最后一个字节后接收到了应答	
		1	发送玩最后一个字节后没有收到应答	
3	DRMI		主机请求数据中断。一旦传输开始，只要它后面没有跟随停止条件，发送器就必须有足够的数据来发送，或者在有数据可发送之前一直将 SCL 线保持为低电平。当主机发送器的数据发送完毕时主机数据请求位就会置位。若主机 TX FIFO 没有数据（即为空）且发送完最后一个字节后没有停止条件标志，那么 SCL 就必须一直保持低电平直至 CPU 写入新的发送字节。当有字节写入主机 TX FIFO 时该位会清零	0
		0	主机发送器不需要数据	
		1	主机发送器需要数据	
4	DRSI		从机请求数据中断。一旦传输开始，只要它后面没有跟随停止条件，发送器就必须有足够的数据来发送，或者在有数据可发送之前一直将 SCL 线保持为低电平。当从机发送器的数据发送完毕时从机数据请求位就会置位。若从机 TX FIFO 没有数据（即为空）且发送完最后一个字节后没有停止条件标志，那么 SCL 就必须一直保持低电平直至 CPU 写入新的发送字节。当有字节写入从机 TX FIFO 时该位会清零	0
		0	从机发送器不需要数据	
		1	从机发送器需要数据	

续上表

位	符号	值	描述	复位值
5	Active		表示总线是否忙碌。该位在起始条件出现时置位，在停止条件出现时清零	0
6	SCL		SCL 信号的当前值	-
7	SDA		SDA 信号的当前值	-
8	RFF		接收 FIFO 满（RFF）。当 RX FIFO 满且不能再接收任何数据时该位置位。当 RX FIFO 不满时该位清零。若在接收 FIFO 满时还有数据达到，则 SCL 保持低电平直至 CPU 读取 RX FIFO 并为该数据腾出空间为止	0
		0	RX FIFO 不满	
		1	RX FIFO 满	
9	RFE		接收 FIFO 空。该位在 RX FIFO 空时置位，在 RX FIFO 有有效数据时清零	1
		0	RX FIFO 有数据	
		1	RX FIFO 空	
10	TFF		发送 FIFO 满。该位在 TX FIFO 满时置位，在 TX FIFO 不满时清零	0
		0	TX FIFO 不满	
		1	TX FIFO 满	
11	TFE		发送 FIFO 空。该位在 TX FIFO 空时置位，在 TX FIFO 有有效数据时清零	0
		0	TX FIFO 有数据	
		1	TX FIFO 空	
31:12	-		保留，用户软件不应向保留位写入 1。从保留位读出的值未定义	NA

13.8.13 I2C控制寄存器（I2C_CTL-0x5000 C308）

I2C_CTL 可用来使能中断和复位 I^2C 状态机。置位时使能的中断会使 USB_I2C_INT 中断输出线被拉高。

表 13.12 I²C 控制寄存器位描述

位	符号	值	描述	复位值
0	TDIE		发送完成中断使能。该位可使能 TDI 中断发信号示意 I²C 发出一个停止条件	0
		0	禁止 TDI 中断	
		1	使能 TDI 中断	
1	AFIE		发送器仲裁失败中断使能。当试图把 SDA 设置为高电平时，该位使能在发送过程中被拉高的 AFI 中断，但总线通过另一个器件被驱动为低电平	0
		0	禁止 AFI	
		1	使能 AFI	
2	NAIE		发送器无应答中断使能。该位可使能 NAI 中断发信号示意发已送的字节未被应答	0
		0	禁止 NAI	
		1	使能 NAI	
3	DRMIE		主机发送器数据请求中断使能。该位可使能 DRMI 中断通知主机发送器已用完数据、尚未发起停止条件，并保持 SCL 线为低电平	0
		0	禁止 DRMI 中断	
		1	使能 DRMI 中断	
4	DRSIE		从机发送器数据请求中断使能。该位可使能 DRSI 中断通知主机发送器已用完数据、尚未发起停止条件，并保持 SCL 线为低电平	0
		0	禁止 DRMI 中断	
		1	使能 DRMI 中断	
5	REFIE		接收 FIFO 满中断使能。该位可使能接收 FIFO 满中断以示意接收 FIFO 不能再接收数据	0
		0	禁止 RFFI	
		1	使能 RFFI	
6	RFDAIE		有可用的接收数据中断使能。该位可使能 DAI 中断来表示接收 FIFO 中有可用数据（即不空）	0
		0	禁止 DAI	
		1	使能 DAI	
7	TFFIE		发送 FIFO 不满中断使能。该位可使能发送 FIFO 不满中断以示意可以向发送 FIFO 写入数据。 注：该 FIFO 不满。这有助于 CPU 向 I²C 模块写数据（FIFO 有空间时），而且不需要轮询状态寄存器	0
		0	禁止 TFFI	
		1	使能 TFFI	

续上表

位	符号	值	描述	复位值
8	SRST		软复位。只有在少许情况下才需要软复位。如：器件发起了起始条件确没发送停止条件。若总线保持忙状态的时间长于超时周期，系统定时器就会复位 I^2C 总线。在软复位时，TX 和 RX FIFO 被刷新，I2C_STS 寄存器清空，所有内部状态机被复位以出现空闲。软复位不能修改 I2C_CLKHI、I2C_CLKLO 和 I2C_CTL（除软复位位以外）	0
		0	请见文本	
		1	将 I^2C 总线复位（闲置）。自清零	
31:9	-	NA	保留，用户软件不应向保留位写入 1。从保留位读出的值未定义	NA

13.8.14 I^2C时钟高电平寄存器（I2C_CLHHI-0x5000 C30C）

CLK 寄存器包含计数 48MHz 时钟周期的最终计数，来创建低速 I^2C 串行时钟 SCL 的高电平周期。

表 13.13 I2C_CLKHI 寄存器

位	名称	功能	复位值
7:0	CDHI	时钟分频器高电平。该值是 48MHz 时钟的数目，串行时钟（SCL）将为高电平周期	0xB9

13.8.15 I^2C时钟低电平寄存器（I2C_CLHLO-0x5000 C310）

CLK 寄存器包含计数 48MHz 时钟周期的最终计数，来创建低速 I^2C 串行时钟 SCL 的低电平周期。

表 13.14 I2C_CLKLO 寄存器位描述

位	名称	功能	复位值
7:0	CDLO	时钟分频器低电平。该值是 48MHz 时钟的数目，串行时钟（SCL）将为低电平周期	0xB9

13.8.16 中断处理

OTGIntSt 寄存器中设置的中断在 HNP 切换过程中置位和清零。所有与 OTG 相关的中断。如果使能，都被连接到 USBIntSt 寄存器中的 USB_OTG_INT 位。

I^2C 相关的中断在 I2C_STS 寄存器中置位，如果使能，则通过 I2C_CTL 连接到位 USB_I2C_INT。

有关设备控制器产生的中断的详细内容请见 USB 设备一章。有关主机控制器产生的中断，请见 OHCI 规范。

USBIntSt寄存器中的EN_USB_INTS位使能连接任何USB相关的中断到NVCI控制器（见图 13.5）。

注：HNP 在主机和设备之间切换过程中（OTG 栈有效），一个操作可将中断的级别提高。建议让 OTG 栈启动基于中断的操作，忽略设备和主机级别的中断。这就意味着，在 HNP 切换过程中，OTG 堆栈为主机和设备控制器提供通信。

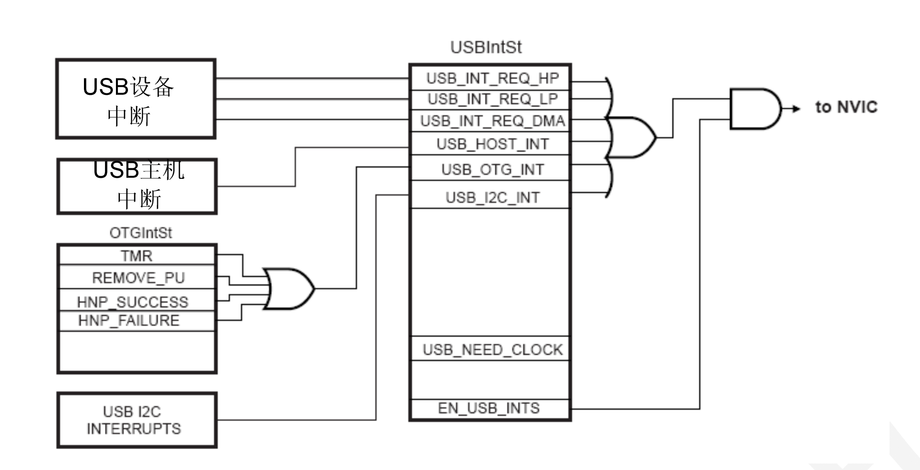


图 13.5 USB OTG 中断处理

13.9 支持HNP

本小节主要介绍 OTG 控制器硬件支持的 HNP 主机交换协议。

当两个双角色设备相互连接时，Mini-AB 插座中的插入的插头类型可确定各设备自身的角色。如果设备的插座插入的是 Mini-A 插头则该设备默认为主机（A-设备），插入的是 Mini-B 插头的设备默认为外设（B-设备）。

一旦连接上，默认主机和默认外设就可以通过主机交换协议自由地切换主从机角色。

OTG控制器的操作流程如图 13.6所示。每个控制器（主机、设备或OTG）都是通过一系列状态和控制寄存器以及中断来和它们的软件栈通信的。OTG软件栈可通过I^2C接口和外部收发器中断信号来和外部OTG收发器进行通信。

OTG 软件栈负责执行 HNP 状态机（按照 USB 2.0 规范的 OTG 补充）。

OTG 控制器提供对某些 HNP 状态机的支持（如下面的小节所述）。

接下来我们将详细介绍 USB 状态机、HNP 切换和 USB 控制器之间通信的内容。

- USB OHCI 规范；
- USB OTG 补充，版本 1.2；
- USB2 0 规范；
- ISP1301 的数据手册和用户手册。

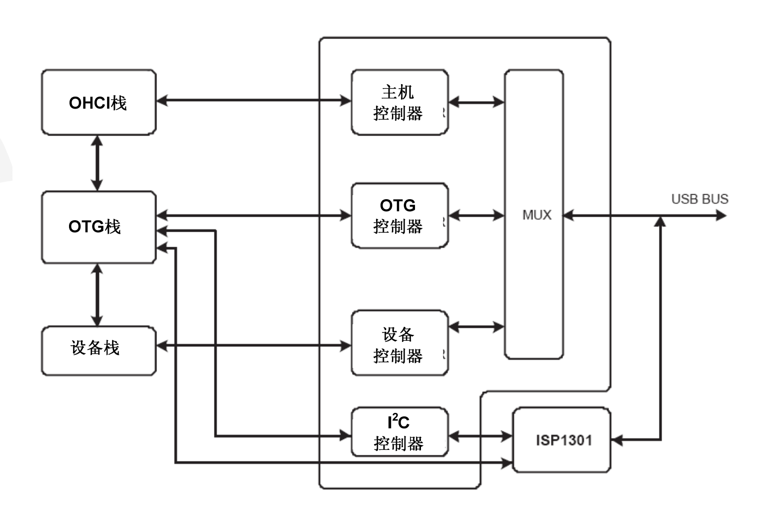


图 13.6 有软件栈的 USB OTG 控制器

13.9.1　B设备从外设切换至主机

在这种情况下，OTG 控制器属 B 设备，默认角色是外设，现在要将其从外设切换为主机。

OTG 补充信息利用一个状态框图定义了双角色 B-设备在 HNP 切换过程中的行为。OTG 软件堆栈负责实现图中所有的状态。

OTG控制器硬件支持双角色B设备状态框图中状态b_peripheral、b_wait_acon和b_host之间的状态转换。置位OTGStCtrl寄存器中的B_HNP_TRACK使硬件支持B设备从外设切换为主机。置位该位后的硬件操作如图 13.7所示。

空闲
B_HNP_TRACK = 0
B_HNP_TRACK = 1 ?
no
总线挂起?
no
断开连接U1的设备控制器
置位REMOVE_PU
PU_REMOVED置位?
yes
检测到总线复位或恢复?
yes
no
重新连接端口U1到设备控制器
PU_REMOVED置位?
no
置位HNP_FAILURE
清零B_HNP_TRACK
和PU_REMOVED
等25μs使总线稳定
检测到与A设备连接?
yes
no
检测到总线复位或恢复?
yes
no
重新连接端口U1到设备控制器
置位HNP_SUCESS
置位PORT_FUNC[0]
驱懂内部主机控制器端口为J状态，驱动U1为SE0
主机发送SE0?
yes
no
连接U1至主机控制器
清除B_HNP_TRACK和
PU_REMOVED

图 13.7　B-设备从外设切换成主机

图 13.8展示了OTG软件堆栈为了响应硬件操作而必须要执行的操作，硬件操作有：置位REMOVE_PU、HNP_SUCCESS和HNP_FAILURE。此外还标明了软件堆栈的操作与双角色B设备状态之间的关系。B设备的状态已用黑体标出，并用椭圆圈住。

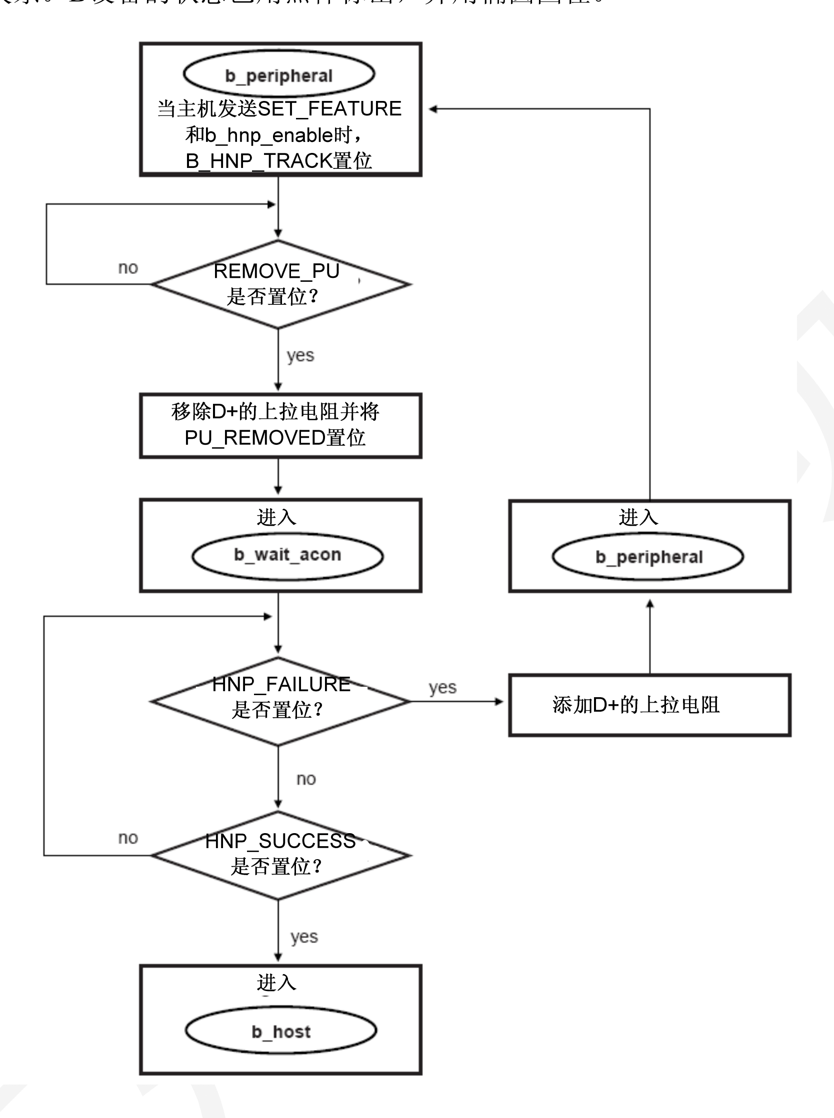


图 13.8 软件的状态转变（外设-主机）

需要注意的是上图只展示了一部分硬件支持的 B 设备的 HNP 状态和状态转变。软件栈是负责执行所有的 HNP 状态的。

从图 13.8来看，似乎应该要轮询中断位REMOVE_PU，但是如果相应的中断已使能就不需要轮询了。

下面给出了完成图 13.8所需操作的代码示例。在该示例中，我们假设外部OTG收发器使用的是ISP1301。

移除 D+的上拉电阻

```
/*通过 ISP1301 移除 D+的上拉电阻*/
OTG_I2C_TX = 0x15A; // 发送 ISP1301 的地址, R/W=0
OTG_I2C_TX = 0x007; // 发送 OTG 控制（清除）寄存器的地址
OTG_I2C_TX = 0x201; // 清零位 DP_PULLUP, 发起停止条件
/*等待 TDI 置位*/
```

```
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
```

增加 **D+**的上来电阻

```
/*通过 ISP1301 增加 D+的上拉电阻 */
OTG_I2C_TX = 0x15A; //发送 ISP1301 的地址,R/W=0
OTG_I2C_TX = 0x006; //发送 OTG 控制（置位）寄存器的地址 s
OTG_I2C_TX = 0x201; // 置位位 DP_PULLUP, 发起停止条件
/*等待 TDI 置位*/
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
```

13.9.2 A-设备：主机切换到外设

在这种情况下，OTG 控制器的角色是 A 设备，默认为主机，现在要将其从主机切换为外设。

OTG 补充信息利用一个状态框图定义了双角色 A-设备在 HNP 切换过程中的行为。OTG 软件堆栈负责实现图中所有的状态。

OTG控制器硬件支持装角色A设备状态框图中状态a_host、a_suspend、a_wait_vfall和a_peripheral之间的转换。置位OTGStCtrl寄存器中的位A_HNP_TRACK使硬件就允许A设备从主机状态切换成设备状态。该位置位后的硬件操作如图 13.9所示。

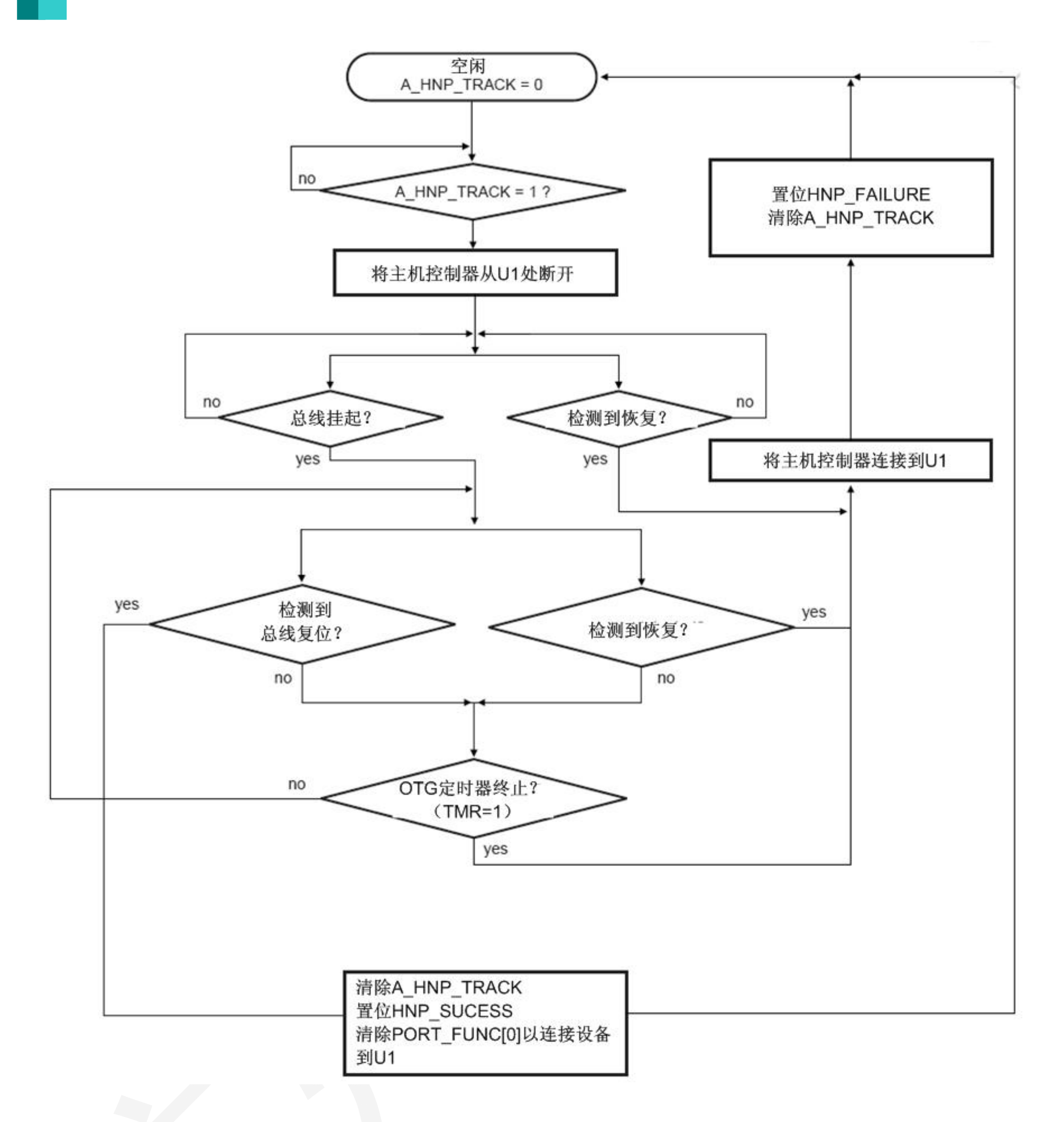


图 13.9　A-设备从主机状态切换到外设状态

图 13.10展示了OTG软件堆栈为了响应硬件操作而应该执行的操作，硬件操作有：置位TMR、HNP_SUCCESS和HNP_FAILURE。此外还标明了软件堆栈的操作与双角色A设备状态之间的关系。A设备的状态已用黑体标出，并用椭圆圈住。

a_host
当主机发送SET_FEATURE
和a_hnp_enable时，
A_HNP_TRACK置位

置位外部OTG收发器中的
BDIS_ACON_EN

装载OTG定时器并使能

端口1上的主机被挂起

进入
a_suspend

TMR是否置
位？
no
yes

HNP_SUCCESS
是否置位？
no
yes

HNP_FAILURE
是否置位？
no
yes

清零外部OTG收发器中的位
BDIS_ACON_EN
Vbus放电

进入
a_wait_vfall

停止OTG定时器

进入
a_peripheral

停止OTG定时器

清零外部OTG收发器中的位
BDIS_ACON_EN

进入
a_host

图 13.10 软件的状态转变（主机-外设）

需要注意的是上图只展示了一部分硬件支持的 A 设备的 HNP 状态和状态转变。软件栈是负责执行所有的 HNP 状态的。

从上图来看，似乎应该要轮询中断位 TMR，但是如果相应的中断已使能就不需要轮询了。

下面给出了完成图 13.10操作所需的代码示例。在该示例中，我们假设外部OTG收发器使用的是ISP1301。

置位外部 OTG 收发器中的位 BDIS_ACON_EN

```
/*置位 ISP1301 中的位 BDIS_ACON_EN */
OTG_I2C_TX = 0x15A; //发送 ISP1301 的地址, R/W=0
OTG_I2C_TX = 0x004; //发送模式控制（设置）寄存器的地址
```

```
OTG_I2C_TX = 0x210; //置位 BDIS_ACON_EN，发起停止条件
/*等待 TDI 置位*/
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
```

将外部 OTG 收发器中的位 BDIS_ACON_EN 清零

```
/*置位 ISP1301 中的位 BDIS_ACON_EN*/
OTG_I2C_TX = 0x15A; //发送 ISP1301 的地址, R/W=0
OTG_I2C_TX = 0x005; //发送模式控制（清零）寄存器的地址
OTG_I2C_TX = 0x210; //清零 BDIS_ACON_EN bit,发起停止条件
/*等待 TDI 置位*/
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
```

V_{BUS} 放电

```
/*将 ISP1301 中的位 VBUS_DRV 清零*/
OTG_I2C_TX = 0x15A; //发送 ISP1301 的地址, R/W=0
OTG_I2C_TX = 0x007; //发送 OTG 控制（清零）寄存器的地址
OTG_I2C_TX = 0x220; //清零 VBUS_DRV,发起停止条件
/*等待 TDI 置位*/
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
/*  将 ISP130 中的位 VBUS_DISCHRG 置位*/
OTG_I2C_TX = 0x15A; //发送 ISP1301 的地址, R/W=0
OTG_I2C_TX = 0x006; //发送 OTG 控制（设置）寄存器的地址
OTG_I2C_TX = 0x240; //置位 VBUS_DISCHRG,发起停止条件
/*等待 TDI 置位*/
while (!(OTG_I2C_STS & TDI));
/*清零 TDI */
OTG_I2C_STS = TDI;
```

装载 OTG 定时器并使能

```
/*假定之前已设置好了 OTG 定时器，定时器的时间刻度为 1ms（TMR_SCALE=“10”，使用的是单触*/
发模式（TMR_MODE=0）。*/
/*装入溢出值来执行 a_aidl_bdis_tmr 定时器*/
/*最小溢出值为 200ms */
/*使能定时器*/
OTG_STAT_CTRL |= TMR_EN;
```

停止 OTG 定时器

```
/*禁能定时器，使得 TMR_CNT 复位为 0*/
OTG_STAT_CTRL &= ~TMR_EN;
/*清除 TMR 中断 */
OTG_INT_CLR = TMR;
```

端口 **1** 上的主机被挂起

```
/*写位 PortSuspendStatus 来将端口 1 的主机挂起*/
/*该示例证明软件需要执行低级别的操作*/
/* The host stack code where this is done will be somewhat more involved. */
HC_RH_PORT_STAT1 = PSS;
```

13.10 时钟和功率管理

OTG控制器的时钟分布如图 13.11所示。

除 ahb_slave_clk 以外，控制器的时钟都由时钟开关控制。当时钟开关使能有效时，其时钟输出就会开启，CLK_ON 输出有效。CLK_ON 信号可在 OTGClkSt 寄存器中观察到。

为了降低功耗，可以将不使用的主机、设备、OTG 和 I^2C 的时钟关闭，方法就是将 OTGClkCtrl 寄存器中对应的时钟使能位清零。当 USB 模块停止运行时，清零寄存器 PCONP 中的位 PCUSB 就可以关闭所有时钟。

如果软件想访问其中一个控制器的寄存器，必须先保证已使能了该控制器的 48MHz 时钟（通过置位 OTGClkCtrl 寄存器中的 CLK_EN），然后再询问 OTGClkSt 中的 CLK_ON 直至它置位为止。一旦 CLK_ON 置位，控制器的时钟就一直保持有效直至软件将 CLK_EN 位清零。在未使能时钟的情况下访问控制器的寄存器会引起数据异常中断。

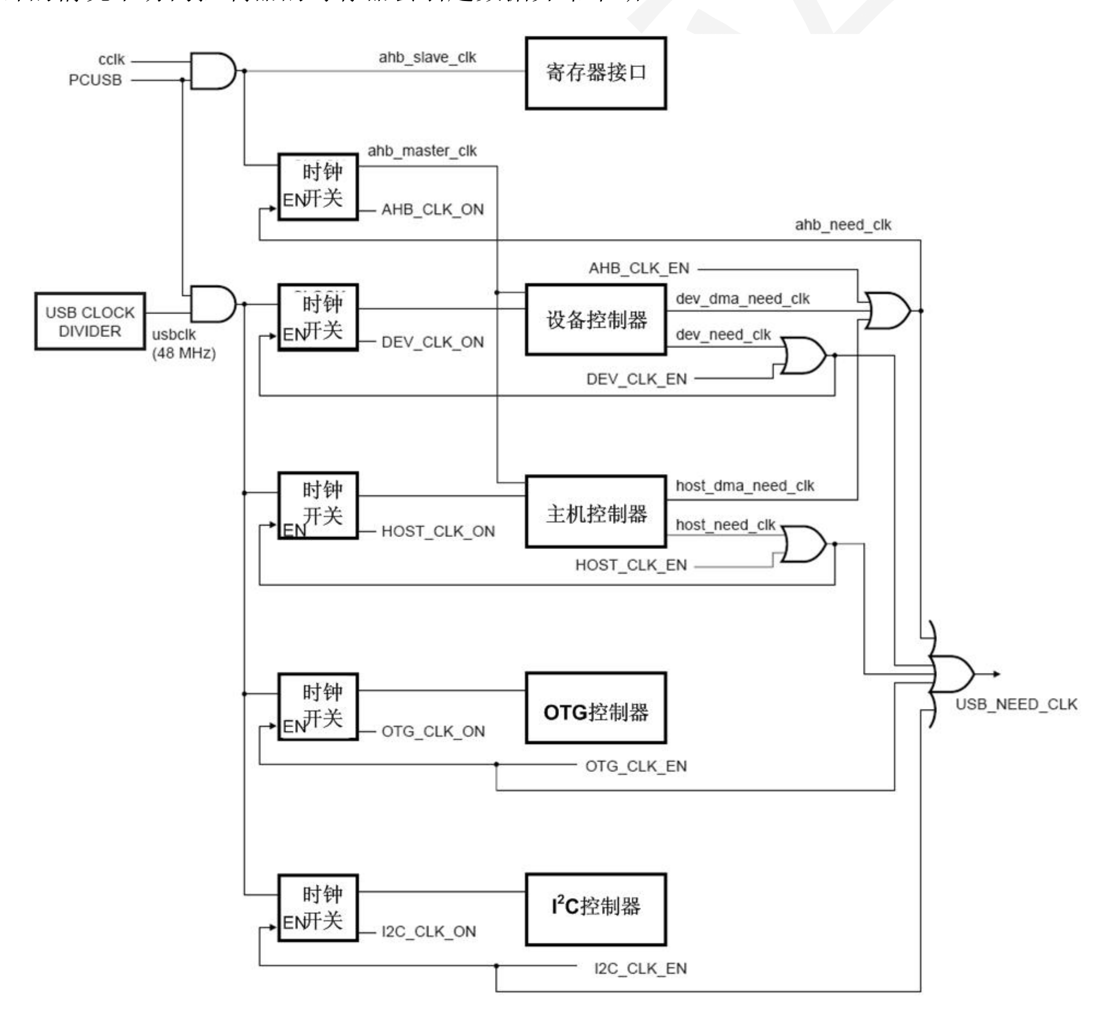


图 13.11 时钟和功率的控制

13.10.1 设备时钟请求信号

设备控制器有两个时钟请求信号：dev_need_clk 和 dev_dma_need_clk。当信号有效时，它们会分别开启设备时钟（48MHz）和 ahb_master_clk。

dev_need_clk 信号在设备未挂起或设备挂起但 USB 总线上仍有数据传输时有效。若发现设备断开连接（清零了 SIE 获取设备状态寄存器中的 CON 位-“SIE 命令代码寄存器”小节），dev_need_clk 无效。当软件不需要访问设备控制器的寄存器时，该信号允许 DEV_CLK_EN 在正常操作中清零。设备会继续照常工作并在设备挂起或断开连接时自动关断时钟。

信号 dev_need_clk 在设备控制器进行 DMA 访问（对存储器）时置为有效。一旦该信号使能，它会保持 2ms（2 帧）的有效状态以保证 DMA 吞吐量不会受到任何延迟（与关闭 ahb_master_clk 有关的）的影响。DMA 访问结束后 2ms，dev_dam_clk 变无效，这有助于降低功耗。该信号允许 AHB_CLK_EN 在正常操作中清零。

主机控制器有两个时钟请求信号：host_need_clk 和 host_dma_need_clk。当信号有效时，它们会分别开启主机时钟（48MHz）和 ahb_master_clk。

host_need_clk 信号在主机控制器不在挂起状态、或在 USBSuspend 并重新发送信号、或 USB 总线上有设备断开时有效。当软件不需要访问设备控制器的寄存器时，该信号允许 DEV_CLK_EN 在正常操作中清零。设备会继续照常工作并在设备挂起或断开连接时自动关断时钟。

信号 dev_need_clk 在设备控制器进行 DMA 访问（对存储器）时置为有效。一旦该信号使能，它会保持 2ms（2 帧）的有效状态以保证 DMA 吞吐量不会受到任何延迟（与关闭 ahb_master_clk 有关的）的影响。DMA 访问结束后 2ms，dev_dam_clk 变无效，这有助于降低功耗。该信号允许 AHB_CLK_EN 在正常操作中清零。

13.10.2 掉电模式支持

LPC1700 系列 Cortex-M3 微控制器可配置成在 USB 总线活动时从掉电模式中唤醒。当芯片掉电且 USB 中断被使能时，USB_NEED_CLK 的有效可使芯片从掉电模式中唤醒。

当 USBWAKE 置位，在可以进入掉电模式之前 USB_NEED_CLK 必须有效。这可以通过清除 OTGClkCtrl 寄存器中的 CLK_EN 位和将主机控制器置于挂起状态来实现。如果要等 dma_need_clk 和 dev_need_clk 其中一个信号变无效，可在 USBIntSt 寄存器中查询 USB_NEDD_CLK 的状态，以确定它们何时都为无效。

13.11 USB OTG控制器初始化

LPC1700 系列 Cortex-M3 微控制器中的 OTG 设备控制器的初始化包含以下几个步骤：

a）通过置位寄存器 PCONP 中的 PCUSB 位来使能设备控制器。

b）配置并使能 USB PLL 或主 PLL 以获得设备时钟 48MHz（USBCLK）和期望的 cclk 频率。为了使设备控制器同步逻辑能正确操作，cclk 的最小频率应为 18MHz。有关 PLL 的设置和配置详情请参见“确定 PLL0 设置程序”小节。

c）通过置位 USBClkCtrl 寄存器的 CLK_EN 位来使能期望的控制器时钟。轮询 USBClkCtrl 寄存器的 CLK_EN 位直至它置位。

d）通过写对应的 PINSEL 寄存器来使能指定的 USB 引脚功能。

e）按照“USB 设备控制器初始化”小节中的步骤对设备控制器进行初始化。

f）按照 OpenHCI 规范中给出的步骤对主机控制器进行初始化。